2С

УДК 004.42

Е.Н. Сипко
Черкасский государственный техологический университет, г. Черкассы, Украина
sipko@mail.ua

Метод последовательного анализа вариантов решения задачи составления расписания занятий

В статье рассматривается проблема решения задачи составления расписания занятий в высших учебных заведениях. Предлагается метод, позволяющий увеличить точность полученных решений задачи составления расписания занятий.

Введение

Организация учебного процесса в высшем учебном заведении является основой подготовки высококвалифицированных специалистов. Базисом учебного процесса обучения является составление «оптимального» расписания учебных занятий. Расписание учебных занятий должно удовлетворять требованиям по обеспечению все более возрастающих требований к будущим специалистам.

Существует достаточное количество программных комплексов для составления учебных расписаний. Первые программы автоматизированного составления расписаний учебных занятий появились в начале 90-х годов. К сожалению, лишь небольшая их часть реально используется в учебных заведениях различного уровня. Главными проблемами остаются: высокая трудоемкость получения решений, несовершенство технологии проектирования расписания, низкая эргономичность процедуры и высокие временные затраты [1].

Проблема составления расписания в любом учебном заведении является в достаточной степени изученной и даже хрестоматийной. Существуют разработанные математические методы решения данной проблемы. Причем единственное приемлемое решение при этом состоит в переборе всех возможных комбинаций конечного расписания с учетом, разумеется, определенных ограничивающих условий. В условиях любого конкретного вуза составление расписания занятий при помощи перебора займет громадное количество машинного времени. Необходимо также учитывать, что при попытке реализовать автоматическое составление расписания не представляется возможным учет всех привходящих факторов, например таких, как пожелания преподавателей о конкретном времени занятий. В противном случае, время составления расписания было бы неопределенно долгим.

Целью данной работы является применение метода последовательного анализа вариантов для уменьшения времени решения задачи составления расписания учебных занятий и увеличения точности выводимых результатов.

Мировой опыт решения задачи составления расписаний

В 90-х годах в нашей стране, равно как в ближнем и дальнем зарубежье, наблюдался бум производства программ составления расписания занятий для персональных компьютеров, которые к этому времени стали массовыми, весьма производительными и, главное, доступными для учебных заведений. Печально, но остается констатировать, что со своей миссией эти программы не справились.

Во-первых, они едва-едва рассчитывали самые простые варианты расписаний. Чуть только появлялись сложности в виде иностранных языков с делением до трех групп, большого количества совместителей, ограничений по аудиториям и прочее, как программы впадали в глубокую задумчивость и выдавали вердикт о невозможности расстановки часов по причине полной несовместимости исходных данных.

Во-вторых, они считали непростительно долго. Часами, а то и сутками. После запуска на расчет пользователь мог спокойно отвлечься на другие дела, а, возвратившись к программе, обнаруживал десятки нерасписанных часов и множество рекомендаций типа «…Вам следует отменить десять методических дней… разрешить преподавателям по пять окон… ввести шестидневку…» и прочее. Даже если программа каким-то чудом выдавала более-менее приемлемое расписание, любая непроизвольная ошибка в исходных данных или спонтанное решение нескольких преподавателей о переносе их методических дней (а это не такая уж редкость), приводили к необходимости считать все снова. Для получения приемлемого результата программу нужно запустить не менее 15 – 20 раз. Умножаем на часы – получаем сутки и недели «автоматического» расчета.

В-третьих, программы 90-х были разработаны под операционную систему MS DOS, в которой очень трудно было создать по-настоящему эргономичный интерфейс. В результате ввод данных мог занимать несколько дней. Интересно, что впоследствии, с переходом на платформу ОС Windows, многие авторы не сочли нужным воспользоваться богатыми возможностями ее графики и механически перенесли свои интерфейсные наработки из DOS в новые программы.

Не очень впечатляющие результаты применения первых программ позволили некоторым разработчикам сформулировать генеральную идею о том, что задача составления расписания занятий лежит вовсе не в плоскости математики, а в области социальных и межличностных отношений. А потому не поддается алгоритмизации в принципе. В результате на рубеже веков на свет появился класс незатейливых программ-редакторов расписаний, которые, не претендуя на глубину и научность, перенесли приемы работы с миллиметровкой, карандашом и ластиком на экран компьютера. При этом за самое главное достижение было выдано то, что программа безошибочно определяет случаи наложения часов преподавателя в разных аудиториях и может показать свободное место, куда можно поставить очередной час.

Суть метода последовательного анализа вариантов

На основе обобщения идей теории последовательных решений и динамического программирования В.С. Михалевич разработал общую схему последовательного анализа вариантов (ПАВ) [2]. С точки зрения формальной логики схема ПАВ сводится к такой последовательности повторения процедур:

– разбиение множества вариантов решения задачи на несколько подмножеств, каждое из которых обладает специфическими свойствами;

– использование этих свойств для поиска логических противоречий в описании отдельных подмножеств;

– исключение из дальнейшего рассмотрения тех подмножеств вариантов решения, в описании которых имеются логические противоречия.

Методика последовательного развития, анализа и отсева вариантов состоит в таком способе развития вариантов и построения операторов их анализа, которые позволяют отсеивать бесперспективные начальные части вариантов до их полного построения.

Поскольку при отсеивании бесперспективных начальных частей вариантов отсеивается тем самым и все множество их продолжений, то происходит значительная экономия вычислительных затрат.

На базе этой общей схемы В.С. Михалевич и его сотрудники разработали целый ряд алгоритмов последовательного анализа вариантов, которые получили широкое применение в практике.

Приближенные методы можно разбить на следующие группы: методы локальной оптимизации, модификации точных методов, эвристические методы, максимально учитывающие специфику решаемых задач, методы случайного поиска, а также методы, сочетающие локальную оптимизацию со случайным поиском.

Отметим, что многие приближенные алгоритмы позволяют решать задачи дискретной оптимизации в диалоговом режиме. Это дает возможность в зависимости от выделяемых ресурсов (времени, памяти компьютера и т.п.) последовательно улучшать полученное решение путем изменения всех или некоторых исходных данных.

Значительная часть приближенных алгоритмов дискретной оптимизации базируется на использовании вычислительных схем известных точных методов, таких как метод ветвей и границ, последовательного анализа вариантов и прочих.

Одними из наиболее развитых приближенных методов являются методы локальной оптимизации, имеющие своей целью отыскание локально оптимальных решений. Нередко в этих методах на определенных этапах решения задачи используются методы случайного поиска, а также различные способы (эвристики), которые дают возможность сократить ход вариантов и максимально учитывают специфику задачи.

Следует отметить, что алгоритмы, в которых комбинируются различные идеи, на практике часто оказываются наиболее эффективными. С помощью этих методов были решены многочисленные сложные задачи классификации объектов, размещения, планирования и проектирования. Основное преимущество этих методов – простота реализации, а основной недостаток – это то, что они не могут адаптироваться к условиям решаемой задачи. Значительно более гибкими являются методы, у которых вероятностный закон зависит от исходов предыдущих испытаний и изменяется от итерации к итерации. Это методы случайного поиска с обучением.

Методы случайного поиска используются для приближенного решения многомерных задач о ранге, а также задач линейного булева программирования большой размерности.

Детализация схемы ПАВ происходила в нескольких направлениях. В связи с решением задач линейной или древовидной структуры был сформирован обобщенный принцип оптимальности для монотонно рекурсивных функционалов, на основе которого можно строить схему решений, свободную от некоторых ограничений, присущих каноническим процедурам динамического программирования.

Для структур, которые имеют форму дерева, были исследованы вопросы, связанные с порядком просмотра ветвей, и найдены способы минимизации числа массивов информации, необходимой для реализации процедур ПАВ.

Метод ПАВ использовался для решения большого количества важных задач оптимального планирования и проектирования, таких как задачи расчета транспортных сетей, задачи размещения на сети типа дерева, проектирование распределительных электрических сетей, выбора оптимальных параметров магистральных газопроводов и т.п.

Далее на основе алгоритмической схемы ПАВ были разработаны эффективные методы конструирования, анализа и отсева вариантов для решения многочисленных задач календарного планирования. Относительно теории календарного планирования

(КП) были обоснованы точные методы решения разных классов задач малой размерности, доказаны необходимые и достаточные условия приоритетной решаемости для задач КП с однотипным оборудованием, разработаны эффективные способы, которые используют правила доминирования.

Для решения задач большой размерности дискретного сепарабельного и линейного целочисленного программирования были разработаны декомпозиционные алгоритмы, которые базируются на схемах последовательного анализа и отсева вариантов. При этом сужение исходного множества вариантов осуществлялось путем отсева отдельных компонент, что дало возможность свести решение исходной задачи большой размерности к решению совокупности подзадач малой размерности.

Метод ПАВ базируется на отсеве бесперспективных элементов как по ограничениям, так и по целевой функции. Его идеи рассмотрим на примере такой задачи дискретного программирования [3].

Схему метода ПАВ изложим для следующей задачи:

$$\min_{x \in D(X)} f(x), \tag{1}$$

при ограничениях

$$g_i(x_1,...,x_n) \le 0, i = \overline{1,m}, \tag{2}$$

$$x_j = Q_j, j = \overline{1,n}, \tag{3}$$

где $Q_j = (q_{1j},...,q_{nj})$ – заданные конечные множества. Вектор $x = (x_1,...,x_n)$ назовем *решением*, если его компоненты $x_j \in Q_j, j = \overline{1,n}$. Множество всех решений обозначим через Ω. Решение называется *допустимым*, если оно удовлетворяет неравенству (2). Множество всех допустимых решений обозначаем через Ω_j. Вектор $x_{(p)} = (x_1,...,x_n)$, $p < n$ будем называть *частичным решением*, если $x_j \in Q_j$. Если при этом он может быть достроен до допустимого решения $(x_1,...,x_p,x_{p+1},...,x_n)$, то будем называть его *допустимым частичным решением*.

Множество всех частичных решений задачи (1) – (3) схематически можно изобразить в виде дерева решений H (рис. 1): вершинам его взаимнооднозначно соответствуют частичные решения, причем висячим вершинам – полные решения $p = n$. Процесс решения задачи с помощью метода ПАВ можно интерпретировать как продвижение на дереве H, начиная с его корня, отвечающего частичному решению $x_{(0)}$.

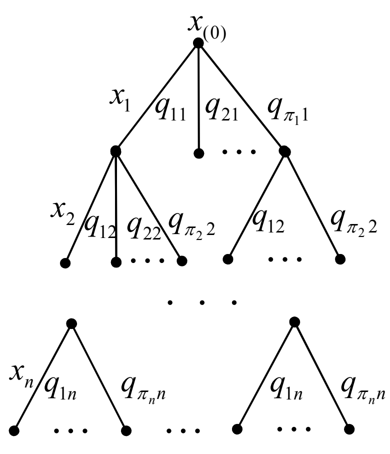


Рисунок 1 – Дерево решений

Чтобы задать алгоритм решения конкретной задачи вида (1) – (3), необходимо указать правило μ выбора частичных решений, подлежащих развитию на каждом шаге, т.е. стратегию последовательного конструирования решений, и построить набор σ элиминирующих тестов ξ_i, осуществляющих отсев частичных решений, которые не могут быть достроены ни до допустимых решений, ни до оптимальных. Естественно, что строить элиминирующие тесты необходимо исходя из специфики решаемой задачи. Предполагаем только, что в набор σ входят тесты ξ_0 – анализа допустимости решений и ξ_1 – сравнения допустимых решений по значению целевой функции. Применяя к любому множеству решений H (содержащему как полные, так и частичные решения) тесты ξ_0 и ξ_1, выявляем в h полные решения и отсеиваем с помощью ξ_0 недопустимые, а с помощью ξ_1 – те из допустимых, для которых в H находятся лучшие по значению целевой функции.

Порядок применения элиминирующих тестов существен с вычислительной точки зрения. Как правило, они упорядочиваются по возрастающей сложности.

Пусть h – некоторое множество частичных решений. Для всякого упорядоченного набора элиминирующих тестов $\sigma = \{\xi_0, \xi_1, ..., \xi_i\}$ введем обозначение

$$\sigma(h) = h^{(t+1)},$$

где

$$h^{(i)} = \frac{h^{(i-1)}}{\xi_{i-1}(h^{(i-1)})}, i = \overline{1, t+1}, h^{(0)} = h.$$

Здесь через $\xi_i(h)$ обозначено множество частичных решений, исключаемых тестом ξ_i. Изложим общий шаг метода последовательного анализа решений.

Пусть в результате k шагов получено множество полных и частичных решений h_k $(h_0 = \{x_{(0)}\})$, которое назовем списком. Во множестве h_k с помощью правила μ выбираем некоторое подмножество $\mu(h_k)$ частичных решений (множество $\mu(h_k)$ может состоять и из одного частичного решения). Каждое из частичных решений $x^0_{(p)} \in \mu(h_k)$ заменяем системой частичных решений вида

$$\beta(x^0_{(p)}) = \{x_{(p+1)} = (x_1, ..., x_{p+1}) | x_j = x^0_j, j = \overline{1, p}, x_{p+1} \in Q_{p+1}\},$$

где оператор β реализует пошаговое конструирование частичных решений. К полученному множеству

$$h^t_{k+1} = \left(\frac{h_k}{\mu(h_k)}\right) \cup \beta(\mu(h_k))$$

применяем набор σ элиминирующих тестов, каждый из которых по определенному признаку осуществляет элиминацию (отсев) бесперспективных частичных решений из множества h^t_{k+1}.

Итак, k-й шаг метода последовательного анализа решений состоит в преобразовании списка h_k по формулам:

$$h^t_{k+1} = \left(\frac{h_k}{\mu(h_k)}\right) \cup \beta(\mu(h_k)), \; h_{k+1} = \sigma(h^t_{k+1}).$$

В силу конечности множества решений и присутствия в наборе σ тестов ξ_0, ξ_1 последовательность $h_0, h_1, ...$ сходится за конечное число шагов к множеству оптимальных решений задачи (1) – (3).

Вариации параметров μ и σ будут приводить к различным алгоритмам метода последовательного анализа решений. Отметим, что если $\sigma = \{\xi_0, \xi_1\}$ или $\sigma = \{\xi_1, \xi_0\}$, т.е. применяются только тесты ξ_0 анализа допустимости и ξ_1 сравнения значений целевой функции на допустимых решениях, то получим семейство алгоритмов полного перебора. В этом случае параметр μ указывает стратегию перебора.

Рассмотрим наиболее типичные способы задания параметра μ, на любом множестве частичных решений h, т.е. укажем правила, по которым на каждом шаге выбираются кандидаты – частичные решения, подлежащие развитию.

Правило 1. $\mu_1(h_k) = h_k$, т.е. на $(k+1)$-м шаге алгоритма развиваются все частичные решения, построенные на предыдущих.

Следующие правила связаны с заданием некоторой функции предпочтения λ.

Правило 2. $\mu_2(h_k) = \{x^*_{(p)}\}$, где частичное решение $x^*_{(p)} \in h_k$ таково, что $\lambda(x^*_{(p)}) \le$ $\le \lambda(x^*_{(s)})$ для всех частичных решений $x^*_{(s)} \in h_k$. Согласно этому правилу, на $(k+1)$-м шаге алгоритма из частичных решений, построенных на предыдущих шагах, развивается тот, на котором достигается минимум функции предпочтения.

Правило 3.

$$\mu_3 = \begin{cases} \mu_2(\Delta h_k), & \text{если} \quad \Delta h_k \ne Щ \min\limits_{x_{(p)} \in h_k} \lambda(x_{(p)}) \le \delta \\ \mu_2(h_k) & \text{в противном} \quad \text{случае} \end{cases},$$

где $\Delta h_k = \dfrac{h_k}{h_{k-1}}$; δ – некоторое неотрицательное число. Основная тенденция процесса с правилом μ_3 – быстрее получить решение, т.е. достичь висячей вершины.

Правило 4. Выбор кандидата на каждом шаге схемы осуществлять из множества Δh_k частичных решений, присоединенных к списку на предыдущем шаге, и только тогда, когда Δh_k не содержит частичных решений, обратиться ко всему списку.

Нетрудно видеть, что правила 2 и 4 являются частными случаями правила 3 при $\delta = 0$ и $\delta = +\infty$.

Правило 5. При выборе кандидата из списка используется функция предпочтения, достигающая минимума на частичном решении, последнем присоединенном к списку. Точнее, пусть множества Δh_k и $\Delta h_{k-s} \bigcap h_k$ для $s = \overline{1, l-1}$ не содержат частичных решений, а $\Delta h_{k-l} \bigcap h_k$ содержит их. Тогда кандидат для k-го шага есть элемент множества $\Delta h_{k-l} \bigcap h_k$.

Одним из основных элиминирующих тестов будет тест ξ_2, основанный на вычислении нижней границы (оценки) для оптимального значения целевой функции $f(x)$ на каждом из множеств $\Omega_f\left(x^0_{(p)}\right)$, элементами которых являются допустимые решения, продолжающие частичное решение $x^0_{(p)}$, т. е.

$$\Omega_f\left(x^0_{(p)}\right) = \Omega_f \bigcap \Omega\left(x^0_{(p)}\right), \text{ где}$$

$$\Omega_f\left(x^0_{(p)}\right) = \left\{ x = (x_1, \ldots, x_n) \,\middle|\, \begin{array}{l} x_j = x^0_j, \; j = \overline{1, p}, \\ x_j \in Q_j, \; j = \overline{p+1, n} \end{array} \right\}$$

множество всех решений, продолжающих $x^0_{(p)}$. *Оценкой* каждого частичного решения $x_{(p)}$ назовем число $\gamma_f(x_{(p)})$, удовлетворяющее соотношению

$$\gamma_f(x_{(p)}) = \min\left\{ f(x) \middle| x \in \Omega_f(x_{(p)}) \right\} \tag{4}$$

(если $\Omega_f(x_{(p)}) = Ш$, то считаем, что правая часть (4) равна $+\infty$).

Пусть f^* – верхняя граница (*рекорд*) для минимума задачи (1) – (3). Для первых шагов схемы можно считать, что рекорд $f^* = +\infty$, а в дальнейшем рекорд равен значению целевой функции на наилучшем найденном допустимом (рекордном) решении.

Если для некоторого частичного решения $x^0_{(p)}$ справедливо $\gamma_f(x_{(p)}) > f^*$, то множество $\Omega_f(x_{(p)})$ не содержит оптимальных решений. Элиминирующий тест ξ_2 для всякого множества частичных решений h задается соотношением

$$\xi_2(h) = \left\{ x_{(p)} \middle| x_{(p)} \in h, \quad \gamma_f(x_{(p)}) > f^* \right\}.$$

Рассмотрим элиминирующий тест ξ_3, основанный на анализе допустимости частичных решений. Пусть для каждой функции $g_i(x)$ существует нижняя граница $\gamma_{g_i}(x^0_{(p)})$ ее значений на множестве $\Omega_f(x^0_{(p)})$. Тогда, если для некоторого l имеет место $\gamma_{g_i}(x^0_{(p)}) > 0$, то частичное решение $x^0_{(p)}$ не может быть достроено до допустимого решения. Для всякого множества частичных решений h элиминирующий тест ξ_3 определим соотношением

$$\xi_3(h) = \left\{ x_{(p)} \middle| x_{(p)} \in h, \quad \max_{1<i<m} \gamma_{g_i}(x_{(p)}) > 0 \right\}.$$

Можно вычислять оценку не для каждой функции $g_i(x)$, а только для некоторой замещающей функции $g(x)$, например для

$$g(x) = \sum_{i=1}^{m} g_i(x).$$

Для задач, в которых достаточно получить одно из оптимальных решений, можно предложить еще один элиминирующий тест ξ_4. Пусть при вычислении $\gamma(x^0_{(p)})$ нашли некоторое решение $x(\gamma) = (x^0_1, ..., x^0_p, x_{p+1}, ..., x_n)$. Если оценка достижима, т.е. а) $x(\gamma \in \Omega_f)$; б) $\gamma(x^0_{(p)}) = f(x(\gamma))$, то частичное решение $x^0_{(p)}$ в множестве h_k заменяется допустимым решением $x(\gamma)$. Последнее в этом случае является лучшим из всех допустимых решений, продолжающих $x^0_{(p)}$.

Применение метода ПАВ для решения задачи составления расписаний занятий в вузе

Рассмотрим задачу составления расписания занятий в вузе [4]:

$$h_1 \cdot \sum_{t \in T} \sum_{p=1}^{P} w(p) \cdot F(p,t) + h_2 \cdot \sum_{t \in T} \sum_{k=1}^{G} D(k,t) \to \min, \tag{5}$$

где h_1 – весовой коэффициент целевой функции «преподавателя»;

t – номер рабочего дня недели;

T – множество номеров рабочих дней для группы k;

p – номер преподавателя;

P – количество преподавателей;

$w(p)$ – весовой коэффициент статуса преподавателя;

$F(p,t)$ – булева переменная;

h_2 – весовой коэффициент целевой функции «студента»;

G – количество групп;

$D(k,t)$ – булева переменная.

Весовые коэффициенты h_1 і h_2 целевых функций «преподавателя» и «студента» определяются с помощью опроса экспертов. Функцию F(p,t) находим из ограничения вида:

$$1 \le M \cdot F(p,t) + Q(p) \le M \quad \forall t \in T; \ \forall p = \overline{1,P}, \tag{6}$$

где M – произвольное положительное достаточно большое число и

$$Q(p) = \sum_{t=1}^{T} \sum_{s=1}^{S} x(p,t,s) \to \min, \tag{7}$$

где s – номер занятия; S – количество занятий;

$$x(p,t,s) = \begin{cases} 1, \text{ если в день t преподаватель p проводит занятие s;} \\ 0, \text{ в противном случае;} \end{cases}$$

Функцию D(k,t) находим из ограничения вида:

$$1 \le M \cdot D(k,t) + R(k) \le M \quad \forall t \in T; \ \forall k = \overline{1,G}, \tag{8}$$

где

$$R(k) = \sum^{T} \sum^{S} y(k,t,s) \to \min; \tag{9}$$

где $y(k,t,s) = \begin{cases} 1, \text{ если в день t в группе k проходит занятие s;} \\ 0, \text{ в противном случае.} \end{cases}$

Функции R(k) и Q(p) – соответственно целевые функции «студента» и «преподавателя». Алгоритм предназначен для нахождения решения с наибольшими значениями этих функций, объединенных в формуле (5). Для решения задачи составления расписания учебных занятий с помощью метода ПАВ вместо формулы (1) используем функцию (5), а далее следуем схеме метода.

Выводы

Применив данный метод для решения задачи составления расписания учебных занятий и взяв в качестве целевой функции интегрированную целевую функцию «преподавателя» и «студента», получили следующие результаты. Предложенный итерационный метод дает возможность получать более точные решения, но при этом время решения задачи увеличивается в среднем в 0,46 раза. Исходя из этого, можно сказать, что метод эффективен для задач небольшой размерности, так как при большой размерности задачи время решения задачи будет существенно увеличиваться.

Литература

1. Веревкин В.И. Автоматизированное составление учебного расписания при проведении занятий циклами / В.И. Веревкин, О.М. Попыхова // Технолого-экономическое образование в XXI веке. Материалы трудов Всероссийской научно-практической конференции (8 – 9 ноября 2007 г.). – 2007. – 38 с.
2. Михалевич В.С. Методы последовательной оптимизации в дискретных сетевых задачах оптимального распределения ресурсов / В.С. Михалевич, А.И. Кукса. – М. : Наука, 1983. – 208 с.
3. Ковалев М.М. Дискретная оптимизация. Целочисленное программирование / Ковалев М.М. – [2-е изд.]. – 2003. – 192 с.
4. Сіпко О.М. Аспекти еволюційної технології для визначення області компромісу між значеннями цільових функцій «викладача» і «студента» у задачі складання розкладів / О.М. Сіпко // Матеріали VIII міжнародної конференції «Інтелектуальний аналіз інформації». – 2007. – С. 128.

О.М. Сіпко

Метод послідовного аналізу варіантів розв'язання задачі складання розкладу занять

У статті розглядається проблема розв'язання задачі складання розкладу занять у вищих навчальних закладах. Пропонується метод, що дозволяє збільшити точність отриманих розв'язків задачі складання розкладу занять.

O.M. Sipko

Method for Sequential Analysis of Variants which Solve the Problem of Scheduling

The article is devoted to the problem of scheduling. The method which increases the accuracy of obtained solutions of the schedule problem is proposed.

Статья поступила в редакцию 01.07.2010.